📦 Opciones de Despliegue - Radio App

Esta guía detalla todas las opciones disponibles para desplegar tu aplicación de radio en producción.

📋 Tabla de Contenidos

📝 Preparación Previa

Antes de desplegar, asegúrate de:

1. Configurar la URL del Backend

Edita `src/environments/environment.prod.ts`:

```
export const environment = {
  production: true,
  apiUrl: 'https://tu-backend-url.com/api',  // ← Cambia esto
  wsUrl: 'wss://tu-backend-url.com'          // ← Para WebSocket (si aplica)
};
```

2. Verificar que la Compilación Funciona

```
npm run build
```

Debe compilar sin errores y generar archivos en `dist/radio-app/`.

3. Probar la Compilación Localmente

```
# Instala un servidor HTTP simple
npm install -g http-server

# Sirve la aplicación compilada
cd dist/radio-app
http-server -p 8080
```

Visita `http://localhost:8080` para verificar.

🖥️ Opción 1: Servidor Propio (VPS)

Ideal para: Control total, tráfico alto, necesidades personalizadas

Requisitos:
- Servidor Linux (Ubuntu 20.04+ recomendado)
- Nginx o Apache
- Dominio propio (opcional)
- Certificado SSL (recomendado)

Paso 1: Preparar el Servidor

```
# Conéctate a tu servidor vía SSH
ssh usuario@tu-servidor.com

# Actualiza el sistema
sudo apt update && sudo apt upgrade -y

# Instala Nginx
sudo apt install nginx -y
```

Paso 2: Compilar la Aplicación

En tu máquina local:

```
# Compila para producción
npm run build -- --configuration production

# Comprime los archivos
cd dist
tar -czf radio-app.tar.gz radio-app/
```

Paso 3: Subir Archivos al Servidor

```
# Opción A: Usando SCP
scp radio-app.tar.gz usuario@tu-servidor.com:/home/usuario/

# Opción B: Usando SFTP
sftp usuario@tu-servidor.com
put radio-app.tar.gz
```

Paso 4: Configurar Nginx

En el servidor:

```
# Descomprime los archivos
cd /var/www
sudo tar -xzf ~/radio-app.tar.gz
sudo mv radio-app html/radio-app

# Crea configuración de Nginx
sudo nano /etc/nginx/sites-available/radio-app
```

Contenido del archivo de configuración:

```
server {
    listen 80;
    listen [::]:80;
    server_name tu-dominio.com www.tu-dominio.com;

    root /var/www/html/radio-app;
    index index.html;

    # Configuración para Angular (SPA)
    location / {
        try_files $uri $uri/ /index.html;
    }

    # Cache para assets estáticos
    location ~* \.(js|css|png|jpg|jpeg|gif|ico|svg|woff|woff2|ttf|eot)$ {
        expires 1y;
        add_header Cache-Control "public, immutable";
    }

    # Compresión gzip
    gzip on;
    gzip_vary on;
    gzip_types text/plain text/css application/json application/javascript text/xml ap
plication/xml application/xml+rss text/javascript;
}
```

```
# Activa el sitio
sudo ln -s /etc/nginx/sites-available/radio-app /etc/nginx/sites-enabled/

# Verifica la configuración
sudo nginx -t

# Reinicia Nginx
sudo systemctl restart nginx
```

Paso 5: Configurar SSL (HTTPS)

```
# Instala Certbot
sudo apt install certbot python3-certbot-nginx -y

# Obtén certificado SSL gratuito
sudo certbot --nginx -d tu-dominio.com -d www.tu-dominio.com

# Configura renovación automática
sudo certbot renew --dry-run
```

Paso 6: Script de Despliegue Automatizado

Crea `deploy.sh` en tu proyecto local:

```
#!/bin/bash

# Configuración
SERVER_USER="usuario"
SERVER_HOST="tu-servidor.com"
SERVER_PATH="/var/www/html/radio-app"

echo "🚀 Iniciando despliegue..."

# Compila
echo "📦 Compilando aplicación..."
npm run build -- --configuration production

# Comprime
echo "🗜️  Comprimiendo archivos..."
cd dist
tar -czf radio-app.tar.gz radio-app/

# Sube al servidor
echo "⬆️  Subiendo al servidor..."
scp radio-app.tar.gz $SERVER_USER@$SERVER_HOST:/tmp/

# Descomprime en el servidor
echo "📂 Desplegando en servidor..."
ssh $SERVER_USER@$SERVER_HOST << 'EOF'
    sudo rm -rf /var/www/html/radio-app/*
    sudo tar -xzf /tmp/radio-app.tar.gz -C /var/www/html/
    sudo systemctl reload nginx
    rm /tmp/radio-app.tar.gz
EOF

# Limpia archivos locales
cd ..
rm radio-app.tar.gz

echo "✅ Despliegue completado!"
echo "🌐 Visita: https://tu-dominio.com"
```

Haz el script ejecutable:

```
chmod +x deploy.sh
./deploy.sh
```

🚀 Opción 2: Vercel (Recomendado)

Ideal para: Despliegue rápido, SSL automático, CDN global

Ventajas:
- ✅ Despliegue automático desde GitHub
- ✅ SSL gratuito
- ✅ CDN global
- ✅ Vista previa de cada commit
- ✅ Plan gratuito generoso

Paso 1: Preparar el Proyecto

Crea `vercel.json` en la raíz del proyecto:

```
{
  "version": 2,
  "builds": [
    {
      "src": "package.json",
      "use": "@vercel/static-build",
      "config": {
        "distDir": "dist/radio-app"
      }
    }
  ],
  "routes": [
    {
      "src": "/(.*)",
      "dest": "/index.html"
    }
  ]
}
```

Agrega script en `package.json`:

```
{
  "scripts": {
    "vercel-build": "ng build --configuration production"
  }
}
```

Paso 2: Desplegar

Opción A: Desde la Web

1. Visita vercel.com (https://vercel.com)
2. Click en “New Project”
3. Importa tu repositorio de GitHub
4. Vercel detectará automáticamente Angular
5. Click en “Deploy”

Opción B: Desde CLI

```
# Instala Vercel CLI
npm install -g vercel

# Inicia sesión
vercel login

# Despliega
vercel --prod
```

Paso 3: Configurar Dominio (Opcional)

En el dashboard de Vercel:

1. Ve a “Settings” → “Domains”

2. Agrega tu dominio personalizado
3. Configura los DNS según las instrucciones

🎯 Opción 3: Netlify

Ideal para: Alternativa a Vercel, excelente para SPAs

Paso 1: Preparar el Proyecto

Crea `netlify.toml` en la raíz:

```
[build]
  command = "npm run build -- --configuration production"
  publish = "dist/radio-app"

[[redirects]]
  from = "/*"
  to = "/index.html"
  status = 200
```

Paso 2: Desplegar

Opción A: Desde la Web

1. Visita netlify.com (https://www.netlify.com)
2. Click en “Add new site” → “Import an existing project”
3. Conecta con GitHub
4. Selecciona tu repositorio
5. Click en “Deploy site”

Opción B: Desde CLI

```
# Instala Netlify CLI
npm install -g netlify-cli

# Inicia sesión
netlify login

# Despliega
netlify deploy --prod
```

📄 Opción 4: GitHub Pages

Ideal para: Proyectos open source, demos

Limitaciones

- ⚠️ Solo sitios públicos (en plan gratuito)
- ⚠️ URL por defecto: `usuario.github.io/repositorio`

Paso 1: Instalar angular-cli-ghpages

```
npm install -g angular-cli-ghpages
```

Paso 2: Configurar Base Href

Edita `angular.json` (o pasa como parámetro):

```
ng build --configuration production --base-href /radio-app/
```

Paso 3: Desplegar

```
# Compila y despliega
ng build --configuration production --base-href /radio-app/
npx angular-cli-ghpages --dir=dist/radio-app
```

Opción: Despliegue Automático con GitHub Actions

Crea `.github/workflows/deploy.yml`:

```yaml
name: Deploy to GitHub Pages

on:
  push:
    branches: [ main ]

jobs:
  build-and-deploy:
    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v3

    - name: Setup Node.js
      uses: actions/setup-node@v3
      with:
        node-version: '18'

    - name: Install dependencies
      run: npm ci

    - name: Build
      run: npm run build -- --configuration production --base-href /radio-app/

    - name: Deploy to GitHub Pages
      uses: peaceiris/actions-gh-pages@v3
      with:
        github_token: ${{ secrets.GITHUB_TOKEN }}
        publish_dir: ./dist/radio-app
```

🔥 Opción 5: Firebase Hosting

Ideal para: Integración con servicios de Google, hosting rápido

Paso 1: Instalar Firebase CLI

```
npm install -g firebase-tools
```

Paso 2: Inicializar Firebase

```
# Inicia sesión
firebase login

# Inicializa el proyecto
firebase init hosting
```

Responde las preguntas:
- **Public directory**: `dist/radio-app`
- **Single-page app**: Yes
- **Set up automatic builds**: No (o Yes si quieres CI/CD)

Paso 3: Configurar firebase.json

```
{
  "hosting": {
    "public": "dist/radio-app",
    "ignore": [
      "firebase.json",
      "**/.*",
      "**/node_modules/**"
    ],
    "rewrites": [
      {
        "source": "**",
        "destination": "/index.html"
      }
    ],
    "headers": [
      {
        "source": "**/*.@(js|css)",
        "headers": [
          {
            "key": "Cache-Control",
            "value": "max-age=31536000"
          }
        ]
      }
    ]
  }
}
```

Paso 4: Desplegar

```
# Compila
npm run build -- --configuration production

# Despliega
firebase deploy --only hosting
```

🔐 Variables de Entorno

Para Desarrollo Local

Crea `.env` (NO subir a Git):

```
API_URL=http://localhost:3000/api
WS_URL=ws://localhost:3000
```

Para Producción

Vercel/Netlify

Configura en el dashboard:
- `API_URL`: URL de tu backend
- `WS_URL`: URL de WebSocket

Servidor Propio

Crea archivo de configuración:

```
# /var/www/config/radio-app.conf
export API_URL=https://api.tu-dominio.com
export WS_URL=wss://api.tu-dominio.com
```

🌐 Configuración de Dominio

DNS Records

```
Type    Name        Value
A       @           IP_DEL_SERVIDOR
A       www         IP_DEL_SERVIDOR
CNAME   api         api.tu-dominio.com
```

Para Vercel/Netlify

Apunta tu dominio a:

Vercel:

```
CNAME   @    cname.vercel-dns.com
```

Netlify:

```
CNAME   @    [tu-sitio].netlify.app
```

🎯 Comparación Rápida

Caracter-ística	VPS	Vercel	Netlify	GitHub Pages	Firebase
SSL Gratis	Con Certbot	✅	✅	✅	✅
CDN Global	Manual	✅	✅	✅	✅
Despliegue Automático	Manual	✅	✅	✅	✅
Control Total	✅	❌	❌	❌	❌
Costo Ini-cial	$5-20/mes	Gratis	Gratis	Gratis	Gratis
Escalabilid-ad	Manual	Automática	Automática	Limitada	Automática
Ideal Para	Producción empresarial	Startups, proyectos modernos	Similar a Ver-cel	Demos, pro-totipos	Apps con Firebase backend

---

🆘 Solución de Problemas

Error: “Failed to load resource”

Causa: URLs del backend incorrectas

Solución:
1. Verifica `environment.prod.ts`
2. Asegúrate de que el backend esté accesible
3. Configura CORS en el backend

Error: “404 Not Found” al recargar

Causa: Configuración de enrutamiento SPA

Solución:
- **Nginx**: Agrega `try_files $uri $uri/ /index.html;`
- **Vercel/Netlify**: Ya está configurado en los archivos anteriores
- **Apache**: Usa `.htaccess` con mod_rewrite

Estilos No Cargan

Causa: Base href incorrecta

Solución:

```
ng build --configuration production --base-href /
```

📚 Recursos Adicionales

- Angular Deployment Guide (https://angular.io/guide/deployment)
- Nginx Documentation (https://nginx.org/en/docs/)
- Vercel Documentation (https://vercel.com/docs)
- Netlify Documentation (https://docs.netlify.com)

🎉 ¡Listo para Desplegar!

Elige la opción que mejor se adapte a tus necesidades y sigue los pasos correspondientes.

Recomendación: Si es tu primera vez desplegando una aplicación Angular, comienza con **Vercel** o **Netlify** para una experiencia sin complicaciones. Una vez que te sientas cómodo, puedes migrar a un servidor propio para mayor control.

¿Necesitas ayuda? Revisa la sección de solución de problemas o consulta la documentación de cada plataforma.